# CSE- 4029 LAB Assignment - 3

**Academic year:** 2021-2022 **Semester:** WIN

**Faculty Name:** Prof. BKSP Kumar raju Alluri **Date:** 18/3/2022

**Student name:** M.Taran **Reg. no.:** 19BCE7346

**Step-1 :** Built the linear regression model and check for VIF(Validation Inflation factor)

```
#Read data
loans_data <-
read.csv("D:/users/lenovo/OneDrive/Desktop/19BCE7346/loans_data.csv",
header=TRUE)

library(dplyr)

glimpse(loans_data)

## Data Preparation

loans_data$Interest.Rate=NA

loans_data=loans_data %>%
mutate(Interest.Rate=as.numeric(gsub("%","",Interest.Rate)) ,
    Debt.To.Income.Ratio=as.numeric(gsub("%","",Debt.To.Income.Ratio)) ,
Open.CREDIT.Lines=as.numeric(Open.CREDIT.Lines) ,
Amount.Requested=as.numeric(Amount.Requested) ,
Amount.Funded.By.Investors=as.numeric(Amount.Funded.By.Investors),
Revolving.CREDIT.Balance=as.numeric(Revolving.CREDIT.Balance)
)

#we will drop variable 'Amount.Funded.By.Investors' since one doesn't have this
information at the time when they come with their loan application.

loans_data = loans_data %>%
```

select(-Amount.Funded.By.Investors)

#Basically FICO is a credit score used by lenders including banks,credit card companies to make decisions about whether or not to offer us a credit.

#substr we will retrieve from 5th till 7th place to create another column f2. After that we will take average of f1 & f2 & keep that values in column f

```
loans_data= loans_data %>%
mutate(f1=as.numeric(substr(FICO.Range,1,3)),
f2=as.numeric(substr(FICO.Range,5,7)),
fico=0.5*(f1+f2)
) %>%
select(-FICO.Range,-f1,-f2)

glimpse(loans_data)
```

#convert employment length to numbers as well where I have used combination of ifelse & substr function to make necessary changes.

```
loans_data=loans_data %>%
mutate(el=ifelse(substr(Employment.Length,1,2)=="10",10,Employment.Length),
el=ifelse(substr(Employment.Length,1,1)=="<",0,el),
el=gsub("years","",el),
el=gsub("year","",el),
el=as.numeric(el)
) %>%
select(-Employment.Length)

glimpse(loans_data)
```

##Employment Length has been removed by el which we created

```
table(loans_data$Loan.Purpose)

round(tapply(loans_data$Interest.Rate,loans_data$Loan.Purpose,mean,na.rm=T))
```

```
loans_data=loans_data %>%
mutate(lp_10=as.numeric(Loan.Purpose=='educational'),
lp_11=as.numeric(Loan.Purpose %in% c("major_purchase","medical","car")),
lp_12=as.numeric(Loan.Purpose %in%
c("vacation","wedding","home_improvement")),
lp_13=as.numeric(Loan.Purpose %in% c("other","small_business","credit_card")),
lp_14=as.numeric(Loan.Purpose %in% c("debt_consolidation","house","moving")))
%>%
select(-Loan.Purpose)

glimpse(loans_data)
```

#Now we are left with categorical variables like Loan.Length, State,Home.Ownership which needs to be converted to dummy variables. Instead of converting one by one Let's write a function for that. We will ignore categories with very low counts.

```
CreateDummies=function(data,var,freq_cutoff=0){
t=table(data[,var])
t=t[t>freq_cutoff]
t=sort(t)
categories=names(t)[-1]
for( cat in categories){
name=paste(var,cat,sep="_")
name=gsub(" ","",name)
name=gsub("-","_",name)
name=gsub("\\?","Q",name)
name=gsub("<","LT_",name)
name=gsub("\\+","",name)
data[,name]=as.numeric(data[,var]==cat)
}
data[,var]=NULL
return(data)
}
```

#Next we will make dummy variables for remaining categorical variables (Loan.Length,State & Home.Ownership) using function CreateDummies.

```
for(col in c("Loan.Length","State","Home.Ownership")){
loans_data=CreateDummies(loans_data,col,100)
}

glimpse(loans_data)

# # Dealing with Missing Values # #
lapply(loans_data,function(x) sum(is.na(x)))

loans_data=loans_data[!(is.na(loans_data$ID)),]

#Now Lets impute the missing values for remaining columns with mean of train
data.We will run a for loop as shown below.

for(col in names(loans_data)){
if(sum(is.na(loans_data[,col]))>0 & !(col %in% c("ID","data","Interest.Rate"))){
loans_data[is.na(loans_data[,col]),col]=mean(loans_data[loans_data$data=="train",c
ol],na.rm=T)
}
}

set.seed(2)
s=sample(1:nrow(loans_data),0.7*nrow(loans_data))
loans_data1=loans_data[s,]
loans_data2=loans_data[-s,]

# variable ID in the modelling process. Its just a id number for the transactions
which should be ignored.
fit=lm(Interest.Rate~. -ID,data=loans_data1)

#we need to drop variables from the model which have high VIF.
library(car)
vif(fit)

# It will be easier to identify high VIF vars if we sort the results and say look at top
3 only as shown below.
```

sort(vif(fit),decreasing = T)[1:3]

## **Step-2 :** Drop the columns based on P values (remove the respective column only when P is < 0.05)

# We'll drop highest vif var one by one and run the model again. First we will drop lp_14 variable as shown below.

fit=lm(Interest.Rate~. -ID - lp_14,data=loans_data)
sort(vif(fit),decreasing = T)[1:3]

# all VIF values are under control i.e below 5.Now We can drop variables with high p-values [ >0.05] one by one or we can use step function which drops vars based on AIC score one by one.

fit=step(fit)

summary(fit)

formula(fit)

#Interest.Rate ~ Amount.Requested + Monthly.Income + Open.CREDIT.Lines +
    Inquiries.in.the.Last.6.Months + fico + Loan.Length_36months +
    State_TX + Home.Ownership_MORTGAGE

##We can drop variable Monthly.Income from this.

fit=lm(Interest.Rate ~ Amount.Requested + Open.CREDIT.Lines +
Inquiries.in.the.Last.6.Months + fico + Loan.Length_36months +
State_TX + Home.Ownership_MORTGAGE,data=loans_data1)
summary(fit)

#This is our final model with all variables being significant( p-value < 0.05).

library(ggplot2)
loans_data1 %>%
mutate(pred_IR=predict(fit,newdata=loans_data1)) %>%

ggplot(aes(x=Interest.Rate,y=pred_IR))+geom_point(alpha=0.6)

## Step-3 : Built the training model again on the common variables

#Create training and test data
#Splitting the dataset in the ratio of 80:20 to create train data and test data.
set.seed(100)  # setting seed to reproduce results of random sampling
trainingRowIndex <- sample(1:nrow(loans_data), 0.80*nrow(loans_data))  # row indices for training data
trainingData <- loans_data[trainingRowIndex, ]  # model training data
testdata  <- loans_data[-trainingRowIndex, ]   # test data

#Fit the model on training data and predict dist on test data
lmMod <- lm(Interest.Rate ~
logAnnual.Income+logAmount.Requested+Inquiries.in.the.Last.6.Months+Amount.Funded.By.Investors+Debt.To.Income.Ratio, data=trainingData)  # build the model
summary(lmMod)
distPred <- predict(lmMod, testData)

## Step-4 : Checking the assumptions of linear regressions(normality assumptions, homoscedasticity, no autocorrelation,..etc)

# Checking model object for actual and predicted values
names(lmModel)
# Histogram to check the distribution of errors
hist(lmModel$residuals, color = "grey")

#checking whether There is any heteroscedasticity
# Using plot function
plot(lmModel)

#There should be no auto serial correlation
library("lmtest")
dwtest(lmModel)

## Step-5 : Test the model performance by RMSE

#Lets look at root mean square error (rmse) value for validation data. We see that model performance [RMSE] on validation data comes out to be

```
rmse= mean((loans_data2$Interest.Rate-predict(fit,newdata=loans_data2))**2)
%>% sqrt()
rmse

#Final Model

fit.final=fit=lm(Interest.Rate ~ .-ID,data=loans_data)
fit.final=step(fit.final)

summary(fit.final)

#Again we will drop Monthly Income from this since it has high value (>0.05)
formula(fit.final)

fit.final=lm(Interest.Rate ~ Amount.Requested  + Open.CREDIT.Lines +
   Inquiries.in.the.Last.6.Months + fico + Loan.Length_36months +
   State_TX + Home.Ownership_MORTGAGE,data=loans_data)
summary(fit.final)
```